

NURBS Surface Reconstruction Using Rational B-spline Neural Networks

Tawfik Elmidany¹, Ahmed Elkeran², Ahmed Galal³, Mohammed Elkhateeb⁴

²Production Engineering and Mechanical Systems Design Department
King Abdul Aziz University
Jeddah, Saudi Arabia

^{1,3,4} Production Engineering and Mechanical Design Department
Mansoura University
Mansoura, Egypt

²elkeran@gmail.com, ⁴elkhateeb@mans.edu.eg

Abstract—Surface reconstruction is a very challenging step in Reverse Engineering. It generates a surface from point cloud acquired from a part surface. NURBS surfaces are commonly used for freeform surface reconstruction. There are several algorithms of NURBS surface reconstruction including: Least Squares Methods, simulated annealing, and particle swarm optimization. The major problem of these methods is that they require parameters and knots optimization which increases the complexity of the problem. The purpose of this paper is to introduce basis function neural networks which eliminates the need for parameters or knots optimization besides providing acceptable error. They are called Rational B-spline Neural Networks (RBNN). Also, they have the advantage of providing the control points and weights of approximated NURBS surface over regular function approximation networks. Training of the network is done using the back-propagation algorithm. Results showed that the RBNN have better approximation to NURBS with acceptable error provided that the number of control points and training rate are selected properly.

Keywords—Freeform Surface, Reverse Engineering, Surface Reconstruction, NURBS, Neural Networks

I. INTRODUCTION

Reverse Engineering (RE) is the process of transforming real parts into digital geometric models. These models can be then modified, analysed, and used for manufacturing these parts using CAD/CAM applications. RE is realized in two steps: surface digitizing and surface reconstruction. Surface digitizing measures the part surface using contact or noncontact techniques. It provides point cloud describing the surface geometry [1,2,3,4].

Surface reconstruction can be defined as the process of recovering the 3D shape of a part. It can be done by: polygonal mesh, or CSG model, or surface fitting. The models produced by polygon mesh are not suitable for CAD/CAM applications as they are not in an engineering drawing format. They are used for rapid prototyping, and animations [1, 5]. Also, the CSG model is suitable only for prismatic shapes and cannot deal with freeform surfaces [15, 21].

Surface fitting is the process of constructing a surface that has the best fit to a series of points. Surface fitting can be done

by interpolation or approximation as shown in fig.1. In interpolation, the resulting surface passes through the point

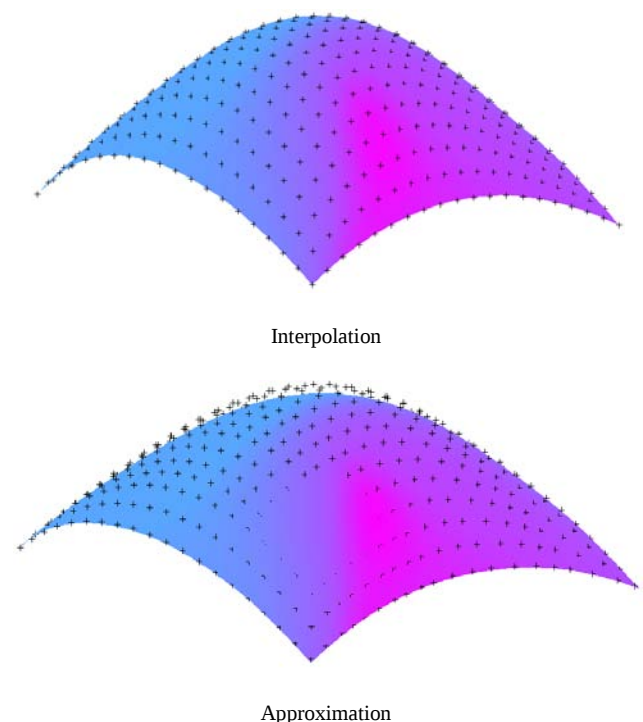


Fig. 1 Difference between Interpolation and Approximation

cloud. This is useful when measurement is done at high precision. In approximation, the produced surface passes as closely as possible to the point cloud. This means approximation is more suitable for point clouds to smooth residual noise presented in the points and to reduce the computational effort required by interpolation.

Parametric surfaces are commonly used for surface fitting. They include Bezier B-Spline and NURBS. NURBS surfaces are preferred in engineering design and manufacturing applications because they permit simple object-shape modification by changing only a small number of parameters, such as control points, knot vectors, and weights [6,7,13].

In this context, the paper presents a new method for NURBS surface approximation without the need of parameters or knots optimization. This method is based on the use of basis function neural networks. They are called Rational B-spline Neural Network (RBNN). They are used due to their higher approximation ability to NURBS surface and they provide the control points and weights which cannot be done by regular function approximation networks. Also, an algorithm is presented for training RBNN based on the back-propagation training algorithm.

II. PREVIOUS WORK

Supposing that there is a point cloud containing $(r \times s)$ $\{Q_{l,k}\}$ points and it is required to approximate these points to a NURBS surface $S(u_{l,k}, v_{l,k})$. Assuming that $S(u_{l,k}, v_{l,k})$ has (p, q) degrees with $(m \times n)$ control points in u and v directions, it can be defined as:

$$S(u_{l,k}, v_{l,k}) = \frac{\sum_{i=1}^m \sum_{j=1}^n W_{ij} P_{ij} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}{\sum_{i=1}^m \sum_{j=1}^n W_{ij} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}$$

where $\{P_{ij}\}$ are control points forming the control grid, $\{W_{ij}\}$ are the weights, $\{N_{i,p}(u)\}$ and $\{N_{j,q}(v)\}$ are the non-rational B-spline basis functions defined over open non uniform knot vectors:

$$\begin{aligned} \bar{U} &= \{0, \dots, 0, \bar{u}_{p+2}, \dots, \bar{u}_{r-p-1}, 1, \dots, 1\} \\ &\quad \begin{matrix} p+1 & p+1 \end{matrix} \\ \bar{V} &= \{0, \dots, 0, \bar{v}_{q+2}, \dots, \bar{v}_{s-q-1}, 1, \dots, 1\} \\ &\quad \begin{matrix} q+1 & q+1 \end{matrix} \end{aligned}$$

where $r = n + p + 1$ and $s = m + q + 1$.

NURBS approximation means determining the control points and weights of $S(u_{l,k}, v_{l,k})$ which minimize the distance between the original points and their corresponding points on $S(u_{l,k}, v_{l,k})$. This can be formulated as a least square expression as following:

$$E = \sum_{l=1}^r \sum_{k=1}^s (Q_{l,k} - S(u_{l,k}, v_{l,k}))^2$$

Before applying approximation methods, a parameterization technique should be used to assign parameter values $(u_{l,k}, v_{l,k})$ to each point. Chord length method is the most common method. It can be formulated as following:

$$\begin{aligned} u_{1,k} &= 0; \quad u_{l,k} = \frac{\sum_{i=2}^l |D_{i,k} - D_{i-1,k}|}{\sum_{i=2}^r |D_{i,k} - D_{i-1,k}|} \\ v_{1,k} &= 0; \quad v_{l,k} = \frac{\sum_{j=2}^k |D_{l,j} - D_{l,j-1}|}{\sum_{j=2}^s |D_{l,j} - D_{l,j-1}|} \end{aligned}$$

After parameterization, knot vectors are to be determined. Knot vectors are used with parameters to calculate the B-spline basis function. To reflect the distribution of parameters, the averaging method is employed to determine the knot vectors. The internal knots can be calculated by:

$$\bar{u}_{j+p+1} = \frac{1}{p} \sum_{i=j+1}^{i+p} u_i$$

Once parameters and knot vectors are calculated, an approximation method is applied to compute the control points and weights of $S(u_{l,k}, v_{l,k})$.

Approximation methods for NURBS surface are divided into two categories: direct methods, and surface skinning methods. Direct methods consist of simultaneously estimating the unknown control points of the surface and weights. Surface skinning, also known as lofting, is a process of passing a smooth surface through a set of so called cross-sectional curves. These curves approximate the measured points. They must have the same degree and knot vectors. This leads to high number of control points making the process more complex and time consuming than direct methods [9,10].

Direct methods include: Least Square Methods (LSM), optimization techniques, and neural networks. The LSM is still the most commonly used method. Wei Yin Ma and J P Kruth [11] used the LSM to approximate measured points to a B-Spline surface. Ma and Kruth [12] used two LSM for NURBS curves and surfaces fitting which automatically identify the control points and their respective weights. The weights of control points are first identified through singular value decomposition, the control points are then determined by least squares minimization. It was found that the surface error produced by LSM was higher. Djordje Brujic et. al. [13] used LSM modified by sparsity structures of the relevant matrices to the problem and regularization terms. This improves the computational complexity and saves time producing lower errors.

As parameterization and knot determination are done by approximated methods, these are reflected in the surface error. To overcome this problem, work has been done in two directions. The first is developing more accurate methods. Aziguli Wulamu et. al. [14] developed a parameterization called "Adaptive Equidistant Parameterization". The surface errors obtained from this algorithm was lower than those of chord length and equidistant methods. Kunal Soni, Daniel Chen, and Terence Lerch [15] used feature based parameterization.

The second direction was to calculate initial parameters and knots by conventional methods, then optimization techniques were then used to refine them at each iteration. Jiing-Yih Lai, Chiou-Yuan Lu [5] used the cumulative chord-length method to assign the initial parameters to the points. Then the Powell's method is applied to optimize the parameters at each iteration until the desired error is reached. Mohammed Riyazuddin [16] used LSM to find the control points of the NURBS surface. Then simulated annealing was applied to optimize the values of weights then the knot vectors.

Meerja Huma [17] used the Simulated Evolution algorithm to optimize the values of knots and weights. Akemi Gálvez et. al. [18] used genetic algorithm to find the parameters of the points. Akemi Gálvez et. al. [19] used two step genetic algorithm. The first is for parameterization and the second for knots determination. Nallig Leal et. al. [20] used evolutionary strategy to obtain the weights of the NURBS surface so that the distance between the point cloud and the NURBS is minimal. Akemi Gálvez and Andrés Iglesias [21] used particle swarm

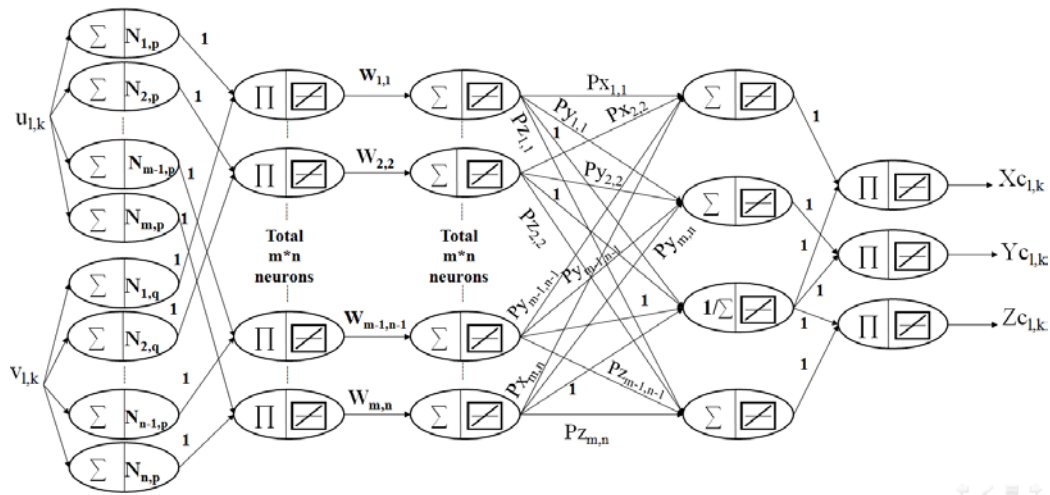


Fig. 2 Architecture of surface approximation RBNN

optimization with LU decomposition to compute parameters, knot vectors, control points, and weights.

Neural networks are used for surface approximation due to their ability to function approximation. Regular function approximation neural networks include: Multi-Layer Feed-Forward (MLF), and Radial Basis Function (RBF). P Gu and X Yan [22] used MLF neural network to approximate parameterized points generated from NURBS surface. Ju H. et. al [4] used RBF neural network to approximate parameterized points acquired by laser scanning freeform surface. After training, these networks can be used only to calculate points on the approximated surface. Although MLF and RBF neural networks have higher approximation ability, they do not provide a solution capable of integration with CAD/CAM applications. This is because they do not provide the control points and weights of the surface which define the parametric surface. Also, using the generated points for data transfer between packages is more difficult and slower than using the parameters of the surface.

To overcome the stated problems, parametric basis function neural networks were developed. They are similar in architecture to MLF neural networks. They have fast approximation ability to approximate points to parametric curves or surfaces due to using parametric basis functions as activation functions. Also, the NURBS control points and weights can be provided after training to reconstruct the surface in CAD/CAM applications, as they are represented in the network by the connecting weights. George K. Knopfa and Jonathan Kofman [7] first used this type of neural networks to approximate points to Bezier surface. Xiaogang G. [8] used B-Spline basis function neural network to approximate points to B-Spline curves and surfaces.

III. RBNN

RBNN are modified B-Spline basis function neural networks which can be used to approximate points to NURBS curves or surfaces. It is noted that the special architectures enable the network inputs and outputs to have the same relationship between the NURBS parameters and coordinates.

There are two architectures of RBNN: the first is for NURBS curve approximation and the second is for NURBS surface approximation.

A. Surface RBNN Architecture

The architecture of surface approximation RBNN is shown in Fig. 2. This architecture has two inputs, parameters $u_{l,k}$ and $v_{l,k}$ of a point, and three outputs which are approximated $X_{C_{l,k}}$, $Y_{C_{l,k}}$, and $Z_{C_{l,k}}$ coordinates of the point. The network has five layers. The number of neurons in the first layer equals to the sum of the number of neurons ($m + n$) in both directions u and v . The second and third layer have $(m \times n)$ neurons, the fourth layer has four neurons, and the fifth layer has 3 neurons. The net input functions are linear sum in the first, third, and fourth layers while the second and fifth have delta function. The activation functions are linear for all layers except the first has B-Spline basis function. The connecting weights between the second and third layers represent the weights of the NURBS, $W_{i,j}$, and they are updated at every training cycle (epoch). The connecting weights between neurons in the third layer and the first, second, and fourth neurons in the fourth layer represent the X , Y , and Z coordinates of the control points ($P_{X_{i,j}} - P_{Y_{i,j}} - P_{Z_{i,j}}$) respectively. These connections are also updated at every epoch. The rest of connections equal unity and are fixed during training. By starting the networks, it performs linear sum of the terms of the networks from the inputs to the outputs. The outputs are calculated by:

$$X_{C_{l,k}} = \frac{\sum_{i=1}^n \sum_{j=1}^m W_{i,j} P_{X_{i,j}} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}{\sum_{i=1}^n \sum_{j=1}^m W_{i,j} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}$$

$$Y_{C_{l,k}} = \frac{\sum_{i=1}^n \sum_{j=1}^m W_{i,j} P_{Y_{i,j}} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}{\sum_{i=1}^n \sum_{j=1}^m W_{i,j} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}$$

$$Z_{C_{l,k}} = \frac{\sum_{i=1}^n \sum_{j=1}^m W_{i,j} P_{Z_{i,j}} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}{\sum_{i=1}^n \sum_{j=1}^m W_{i,j} N_{i,p}(u_{l,k}) N_{j,q}(v_{l,k})}$$

B. Training

During training, the connecting weights of the network are iteratively adjusted to minimize the network performance function. The performance function for the network is the mean square error between the network outputs and the target outputs. The outputs are the approximated $X_{c,l,k}$, $Y_{c,l,k}$, and $Z_{c,l,k}$ coordinates of the points, while the target outputs are the measured $X_{t,l,k}$, $Y_{t,l,k}$, and $Z_{t,l,k}$ coordinates of the points. For $(r \times s)$ points, the performance function can be calculated by:

$$E = \sum_{l=1}^r \sum_{k=1}^s \left((X_{t,l,k} - X_{c,l,k})^2 + (Y_{t,l,k} - Y_{c,l,k})^2 + (Z_{t,l,k} - Z_{c,l,k})^2 \right)$$

By applying the back-propagation training algorithm, the training rules for epoch $k+1$ can be expressed as:

$$\begin{aligned} W_{i,j}(k+1) &= W_{i,j}(k) - \eta \left(\frac{\partial E}{\partial W_{i,j}} \right) \\ P x_{i,j}(k+1) &= P x_{i,j}(k) - \eta \left(\frac{\partial E}{\partial P x_{i,j}} \right) \\ P y_{i,j}(k+1) &= P y_{i,j}(k) - \eta \left(\frac{\partial E}{\partial P y_{i,j}} \right) \\ P z_{i,j}(k+1) &= P z_{i,j}(k) - \eta \left(\frac{\partial E}{\partial P z_{i,j}} \right) \end{aligned}$$

where η represents the training rate. The training rate is held constant throughout the training. The performance of the algorithm is very sensitive to the proper setting of the training rate. If the training rate is set too high, the algorithm may oscillate and become unstable. If the training rate is too low, the algorithm will take too long to converge. To obtain a better training process, an algorithm was developed. Fig. 3 shows the flow chart of the algorithm followed for training the network to reach the desired error. This algorithm starts by using small training rate and number of control points. Then they are gradually increased at each epoch until reaching the desired error.

IV. CASE STUDY

To validate the developed algorithm, it was applied on 600 points, shown in Fig. 5-a, obtained from B-spline surface. These points were to be fitted to a bicubic NURBS surface. At first, the network was constructed and the initial number of control points was assumed to be (4×4) . The initial value of training rate was set to (0.01). By training the network, it was found that the mean square error was high. By increasing the number of control points and the training rate, the mean square error began to drop. When the number of control points was (11×11) and the training rate was (0.1169), a mean square error of $(15.6 \times 10^{-3} \text{ mm})$ was reached. This error was considered as a suitable error according to the number of points and the large area of the surface form which points were obtained. The approximated NURBS surface is shown in Fig. 4-b. Plot of the training process is shown in Fig. 5. Training was done in 100 epochs in 18 seconds. The code was written in MATLAB.

V. CONCLUSION

Freeform surface reconstruction is a major challenge in Reverse Engineering. The goal of surface reconstruction is to approximate point cloud acquired from the surface to a suitable surface model with acceptable error. NURBS surfaces are widely used in the approximation of point clouds for freeform surfaces. NURBS approximation methods are used to compute the control points and weights of the approximated surface. The problem of most of these methods is that they require parameters and knots optimization to provide more enhancements to reduce the approximation error. This is because the methods of calculating knot vectors and parameters are approximated methods. Optimization adds more complexity to the process and makes it more difficult to solve. In this paper, RBNN are used for NURBS surface approximation. Compared to other NURBS approximation methods, they do not require parameters or knots optimization. This is due to the high ability of function approximation of neural networks. Also, after training RBNN, they provide the control points and weights of the NURBS surface which is not available in regular function approximation networks. The performance of the network depends on two factors: the training rate and the number of control points. It is found that the optimum value of training rate depends on the volume of point clouds, and the number of control points. Large volume point clouds require higher training rate while smaller require smaller training rate. Also, increasing the number of control points should be accompanied with an increase in the training rate. Therefore, the values of the number of control points and training rate should be balanced together to achieve optimum performance for the network.

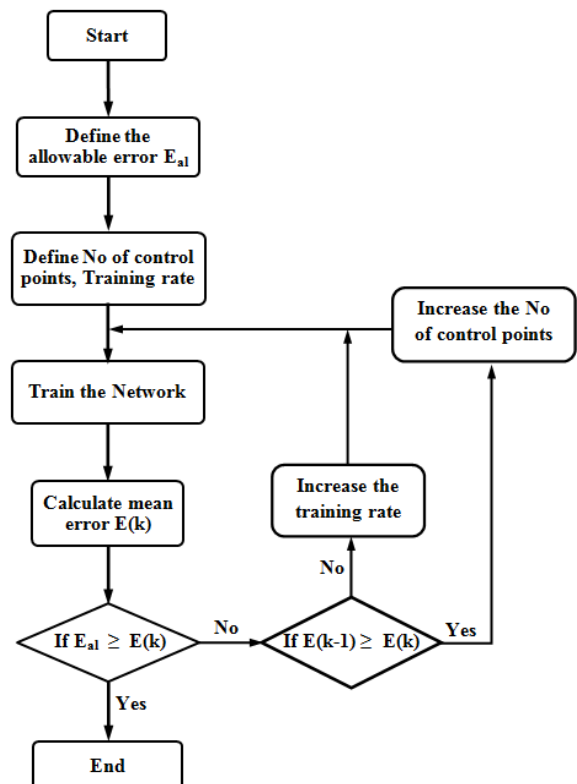


Fig. 3 Flow chart of the algorithm followed for training RBNN to reach the desired error

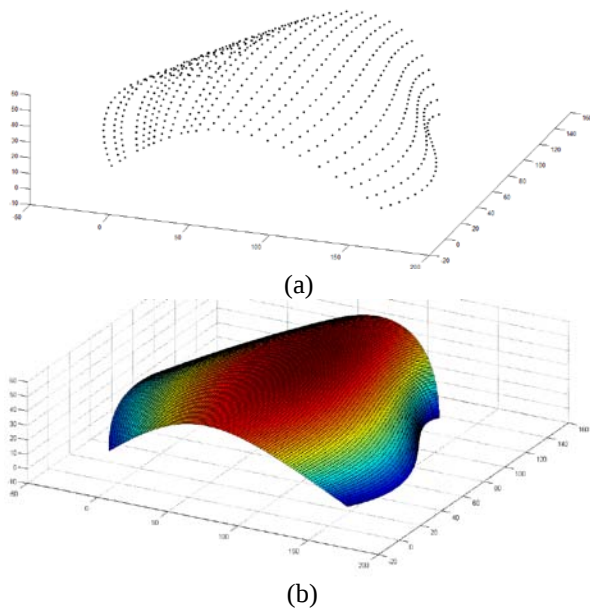


Fig. 4 (a) 600 regular data point (b) Approximated NURBS surface.

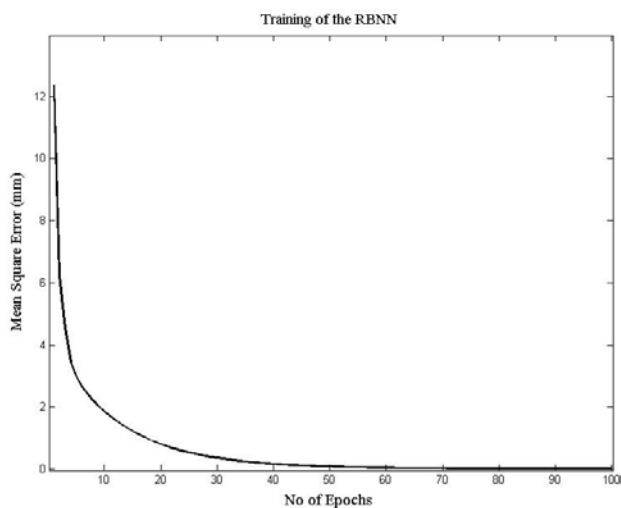


Fig. 5 Training of the network

REFERENCES

- [1] Vinesh Raja and Kiran J. Fernandes, "Reverse Engineering An Industrial Perspective", Springer Series in Advanced Manufacturing. 2008.
- [2] E. Savio, L. De Chiffre, R. Schmitt, "Metrology of freeform shaped parts", CIRP Annals, Vol. 56 No.2, PP810-835, 2007.
- [3] J.-P. Pernot, B. Falcidieno, F. Giannini, J.-C. Leon, "Incorporating free-form features in aesthetic and engineering product design: State-of-the-art report", Computers in Industry, Vol. 59 No.7, PP626-637, 2008.
- [4] Ju Hua, Wang Wen, Xie Jin, Chen Zi-chen, "Neural Network Approach for modification and fitting of digitized data in Reverse Engineering", Journal of Zhejiag University Science, Vol. 5 No. 1, PP75-80, 2004.
- [5] Jiing-Yih Lai, Chiou-Yuan Lu, "Reverse Engineering of Composite Sculptured Surfaces", Int J Adv Manuf Technol, Vol. 12, PP180-189, 1996.
- [6] Pramod N Chivate, Andrei G Jablokow, "Review of surface representations and fitting for reverse engineering", Computer Integrated Manufacturing Systems, Vol. 8 No. 3, PP193-204, 1998.
- [7] George K. Knopfa, Jonathan Kofman, "Adaptive reconstruction of free-form surfaces using Bernstein basis function networks", Engineering Applications of Artificial Intelligence, Vol. 14 No. 5, PP577-588, 2001.
- [8] Xiaogang Guo, "Reconstruction of parametric curves and surfaces using an adaptive basis function network", M.Sc Thesis, Faculty of Graduate Studies, The University of Western Ontario, 1998.
- [9] K. R. Koch, "Identity of simultaneous estimates of control points and of their estimates by the lofting method for NURBS surface fitting", Int J Adv Manuf Technol, Vol. 44 No. 11-12, PP1175-1180, 2009.
- [10] S.M.Shamsuddin, M.A.Ahmed, Y.Samian, "NURBS skinning surface for ship hull design based on new parameterization method", Int J Adv Manuf Technol, Vol. 28 No. 9-10, PP 936-941, 2006.
- [11] Weiyin Ma, J P Kruth, "Parameterization of randomly measured points for least squares fitting of B-Spline curves and surfaces", Computer-Aided Design, Vol. 27 No.9, PP663-675, 1995.
- [12] Ma W, Kruth JP, "NURBS curve and surface fitting for reverse engineering", International Advanced Manufacturing Technology, Vol 14 No. 12, PP918-27, 1998.
- [13] Djordje Brujic, Iain Ainsworth, Mihailo Ristic, "Fast and accurate NURBS fitting for reverse engineering", Int J Adv Manuf Technol, Vol. 41 No. 9-10, PP 948-959, 2009.
- [14] Aziguli Wulamu, Goetting Marc, Zeckzer Dirk, "Approximation of NURBS Curves and Surfaces Using Adaptive Equidistant Parameterizations", Tsinghua Science And Technology, Vol. 10 No. 3, PP 316-322, 2005.
- [15] Kunal Soni, Daniel Chen, Terence Lerch, "Parameterization of prismatic shapes and reconstruction of free-form shapes in reverse engineering", Int J Adv Manuf Technol, Vol. 41 No. 9-10, PP 948-959, 2009.
- [16] Mohammed Riyazuddin, "Visualization with NURBS using simulated annealing optimization technique", M.Sc Thesis, King Fahd University Of Petroleum and Minerals, 2004.
- [17] Meerja Huma, "Evolutionary heuristic optimization for digital curves and surfaces", M.Sc Thesis, King Fahd University Of Petroleum and Minerals, 2005.
- [18] Akemi Gálvez, Andrés Iglesias, Angel Cobo, Jaime Puig-Pey, "Jesús Espinola. Bézier Curve and Surface Fitting of 3D Point Clouds Through Genetic Algorithms, Functional Networks and Least-Squares Approximation", Lecture Notes in Computer Science, Vol. 4706, PP680-693, 2007.
- [19] Akemi Gálveza, Andrés Iglesias, Jaime Puig-Peya, "Iterative two-step genetic-algorithm-based method for efficient polynomial B-spline surface reconstruction", Information Sciences, In Press, Available online 8 October 2010.
- [20] Nallig Leal, Esmeide Leal, and John William Branch, "Simple Method for Constructing NURBS Surfaces from Unorganized Points", Proceedings of the 19th International Meshing Roundtable 2010, Part 3, PP 161-175, 2010.
- [21] Akemi Gálvez, Andrés Iglesias, "Particle swarm optimization for non-uniform rational B-spline surface reconstruction from clouds of 3D data points", Information Sciences, In Press, Available online 13 November 2010.
- [22] P Gu, X Yan, "Neural network approach to the reconstruction of freeform surfaces for reverse engineering", Computer-Aided Design, Vol. 27 No.1, PP 59-64, 1995.